**Sàng số nguyên tố (Sieve of eratosthenes) tiếp theo.**

**Phần I.***(Hai bạn Hiếu và Phúc làm bài tập phần này)*

**Bài 1:** **Prime Subtraction**

**time limit per test: 2 seconds**

Bạn được cho hai số nguyên **x** và **y** *(đảm bảo x > y)*. Bạn được chọn bất cứ số nguyên tố **p** nào và trừ đi vô số lần tùy ý từ số **x** đã cho. Nếu từ số **x** có thể thay đổi thành số **y** hay không?.

Nhắc lại số nguyên tố là số nguyên dương có chính xác hai ước là **1** và chính nó. Chuỗi số nguyên tố như **2, 3, 5, 7, 11 …**

**Input**

* Dòng đầu tiên chứa một số nguyên **t**– Số lượng trường hợp.
* Sau đó **t** dòng, mỗi dòng tương ứng với một trường hợp. Trên mỗi dòng đó chứa 2 số nguyên **x** và **y**

**Output**

Trong mỗi trường hợp, in ra “**YES**” nếu có khả năng chọn một số nguyên tố **p** và trừ đi nhiều lần từ **x** thu được **y**. Ngược lại in ra “**NO**”.

**Example**

**input**

4

100 98

42 32

1000000000000000000 1

41 40

**output**

YES

YES

YES

NO

**Giải thích**

Trường hợp đầu tiên của ví dụ, bạn nên chọn **p** **=** **2** và trừ đi **một lần.**

Trường hơp thứ hai của ví dụ, bạn nên chọn **p** **=** **5** và trừ đi **hai lần**. (*Chú ý bạn không thể chọn* ***p******=******7****, trừ nó, sau đó chọn****p******=******3****trừ thêm lần nữa)*

Trong trường hợp thứ 3, bạn nên chọn **p = 3** và trừ nó đi **333333333333333333** lần.

**Bài 2: Reltively Prime Pais**

**time limit per test: 2 seconds**

Bạn được cung cấp tất cả tập hợp các số nguyên từ L đến R, và (R – L) luôn là số lẻ.

Bạn muốn chia những số đó thành chính xác cặp. Theo cách mà GCD (*ước số chung lớn nhất tiếng anh là greatest common divisor* ) của **i** và **j** bằng **1**. Mỗi số xuất hiện đúng một lần trong 1 cặp.

**Giải thích ước số chung lớn nhất:**

**VD:** **\_\_gcd**( 12 , 24 ) = 12.

**24 = 1\*2\*2\*2\*3** và **12 = 1\*2\*2\*3** nên cả **24** và **12** đều chia hết cho **12**. Và **12** là ước lớn nhất của **12** với **24**.

Code mẫu **\_\_gcd( x, y )** với độ phức tạp O( log ( max ( a, b ))):

int \_\_gcd(long long a, long long b)

{

if (b==0)

return a;

return \_\_gcd( b , a%b );

}

Tham khảo: <https://www.geeksforgeeks.org/c-program-find-gcd-hcf-two-numbers/>

**Input**

* Một dòng duy nhất chứa 2 số nguyên L and R với

**Output**

* Nếu tồn tại đáp án, print "**YES**" ở dòng đầu tiên. Mỗi dòng tiếp theo củacặp  phải bao gồm một cặp số nguyên. **GCD** của mỗi cặp phải bằng **1**. Tất cả (R−L+1) số phải khác nhau và có giá trị nằm trong khoảng từ **L** đến **R**.
* Nếu có nhiều đáp án thỏa mãn, in bất cứ đáp án nào.
* Nếu không tồn tại đáp án, print "**NO**".

**Example**

**input**

1 8

**output**

YES

2 7

4 1

3 8

6 5

**Bài 3: Noldbach problem**

**time limit per test: 2 seconds**

Anh Duy thích những số nguyên tố. Do vậy anh muốn đưa ra một bài toán đố Phúc và Hiếu. Bài toán của anh cho như sau:

Anh giả sử rằng mỗi số nguyên lớn hơn 2 có thể được biểu diễn là tổng của hai số nguyên tố. Bài toán của anh cho rằng có ít nhất **k** số nguyên tố từ 2 đến **n** khác nhau có thể biểu diễn thành tổng của ba số nguyên *(trong đó* ***hai số nguyên tố liền kề*** *và số* ***1****)* .

* Ví dụ:

Hai số nguyên tố được gọi là liền kề nếu giữa chúng **không tồn tại** một số nguyên tố nào khác.

**Input**

Dòng đầu tiên của Input chứa hai số nguyên ***n*** (2 ≤ *n* ≤ 1000) và ***k*** (0 ≤ *k* ≤ 1000).

**Output**

Đầu ra **“YES**” nếu có ít nhất ***k*** số nguyên tố từ 2 đến ***n*** khác nhau có thể biểu diễn như mô tả ở phía trên. Ngược lại thì in ra “**NO**”.

**Examples**

**input**

27 2

**output**

YES

**input**

45 7

**output**

NO

**Giải thích**

Trong ví dụ đầu tiên câu trả lời là **YES** vì ít nhất 2 số có thể biểu diễn (ví dụ, **13** và **19**). Trong ví dụ thứ hai câu trả lời là **NO** vì không thể biểu diễn **7** số nguyên tố từ **2** đến **45** trong dạng mong muốn.

**Phần II**. *(Bạn* ***Thành*** *làm bài tập phần này)*

**Bài 1. Số Phong Phú**

**Time limit 0.38s**

Trong số học, số phong phú là các số mà tổng các ước số của số đó ( không kể chính nó) lớn hơn số đó. Ví dụ, số **12** có tổng các ước số (không kể **12** ) là **1 + 2 + 3 + 4 + 6 = 16** > **12**. Do đó, **12** là một số phong phú.

Bạn hãy lập trình đếm xem có bao nhiêu số phong phú trong đoạn

#### **Input**

* Gồm **2** số **L, R**

#### **Output**

* Gồm **1** số nguyên duy nhất là số phong phú trong đoạn

#### **Giới hạn**

* Có **50%** số test có

#### **Sample Input**

1 50

#### **Sample Output**

9

#### **Note**

* Từ **1** đến **50** có **9** số phong phú là: **12, 18, 20, 24, 30, 36, 40, 42, 48.**

**Bài 2: Different Divisors**

**Time limit per test: 1 second**

Số nguyên dương **x** gọi là số chia của số nguyên **y**, nếu **y** chia hết cho **x** dư 0. Ví dụ, **1** là số chia của **7** và **3** không phải số chia của **8**.

Bạn có một số nguyên dương **d** và yêu cầu bạn tìm số nguyên dương nhỏ nhất **a**, như sau:

* **a** có ít nhất **4** số chia
* sự khác nhau giữa hai số chia của **a** ít nhất là **d**.

**Input**

* Dòng đầu tiên chứa một số nguyên **t** (1 ≤ **t** ≤ 3000) — số lượng trường hợp.
* Dòng đầu tiên của mỗi trường hợp chứa một số nguyên **d** (1 ≤ **d** ≤ 10000).

**Output**

Mỗi một trường hợp in ra một số nguyên **a** — đáp án cho trường hợp đó.

**Example**

**input**

2

1

2

**output**

6

15

**Note**

Trong trường hợp đầu tiên, số nguyên **6** có những số chia sau: **[****1****,****2****,****3****,****6****]**. ở đó có **4** số được liệt kê và sự khác biệt bất kỳ hai trong số chúng ít nhất là **1**. Không có số nguyên nào nhỏ hơn có ít nhất **4** số chia.

Trong trường hợp thứ **2**, số nguyên**15** có những số chia sau:**[****1****,****3****,****5****,****15****]**. ở đó có **4** số được liệt kê và sự khác biệt bất kỳ hai trong số chúng ít nhất là **2**.

Câu trả lời là **12** không hợp lệ vì số chia là **[****1****,** **2****,** **3****,** **4****,** **6****,** **12****].** Và sự khác nhau trong số chúng, Ví dụ, số chia **2** và **3** có hiệu bằng **1** nhỏ hơn **d** là **2**.

**Bài 3: Pleasant Pairs**

**time limit per test: 2 seconds.**

Bạn được cho một mảng số nguyên bao gồm **n** số nguyên riêng biệt. Đếm số cặp của vị trí mà và

**Input**

* Dòng đầu tiên chứa số nguyên **t**  — số lượng những trường hợp.
* Dòng đầu tiên của mỗi trường hợp bao gồm một số nguyên **n**  — độ dài của mảng **a**.
* Dòng thứ hai của mỗi trường hợp bao gồm **n** khoảng trống riêng biệt của **n** số nguyên   — mảng **a**. Luôn đảm bảo rằng mọi phần từ là **khác biệt**.
* Đảm bảo rằng tổng của **n** trên tất cả các trường hợp thử nhiệm không vượt quá

**Output**

Cho mỗi một trường hợp, in ra số lượng cặp của vị trí mà và

**Example**

**input**

3

2

3 1

3

6 1 5

5

3 1 5 9 2

**output**

1

1

3

**Note**

Cho trường hợp đầu tiên, cặp thỏa mãn điều kiện là **(1, 2)**, và

Cho trường hợp thứ hai, cặp thỏa mãn duy nhất là **(****2****,****3****)**.

Cho trường hợp thứ ba, các cặp thỏa mãn điều kiện là **(****1****,****2****),****(****1****,****5****),** và **(****2****,****3****)**.